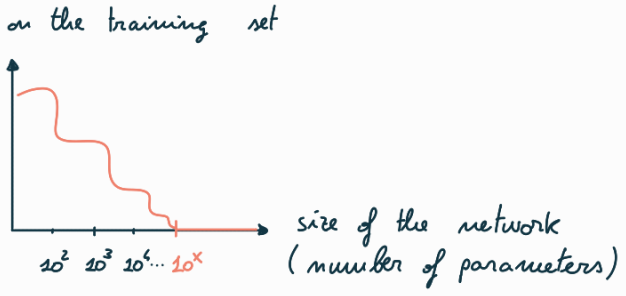
Regularization

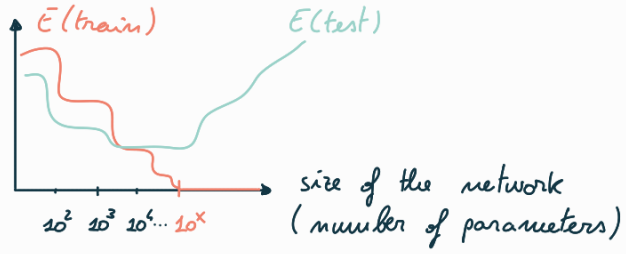
There are some studies that says that you can find a network big enough to guarantee that the error will be zero



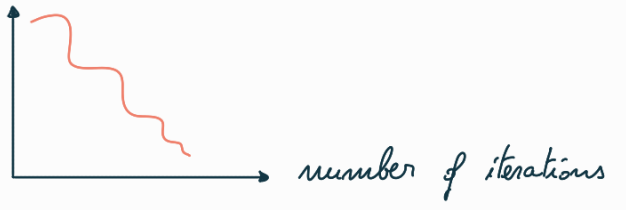
10^x for this particular training set will guarantee error = zero

We have a different situation during testing

This means that when the error during training is zero we are perfectly shaping the dataset ⟶ we have overfitting



The same thing can also happen with respect to the number of iterations. In this case we won't get error (training) = 0 but the error on training will always decrease.



Regularization is an important feature to reduce overfitting.
For feedforward neural networks we have several options that can even be applied together:

1. Parameter norm penalties
2. Dataset augmentation
3. Early stopping
4. Parameter sharing
5. Dropout

① Parameter norm penalties

Add a regularization term E_{reg} to the cost function (as seen in regression)

$$E_{reg}(\theta) = \sum_j |\theta_j|^q$$

Resulting cost function: $\tilde{J}(\theta) = J(\theta) + \lambda E_{reg}(\theta)$

λ is a parameter that weights how much we want to regularize. If λ is very small the regularization effect will be negligible, if λ is too big then the regularization term might be too strong and we could have underfitting

$$E_{reg}(\theta) = \sum_j |\theta_j|^q \begin{cases} q = 1 \longrightarrow \text{manhattan distance} \\ q = 2 \longrightarrow \text{euclidean norm} \\ \dots \end{cases} \Big\} \text{most common regularization terms}$$

argmin $E_D(w) + \lambda E_W(w)$
Linear Regression with regularization is called Ridge Regression

A frequently used regression term in Ridge Regression is

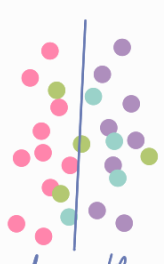
$$E_W(w) = \frac{1}{2} w^T w$$

② Dataset augmentation

Generate additional data and include it in the dataset.
The overfitting is due to the ability of the network to perfectly shape the dataset.
To avoid this we could add noise or data transformations to the dataset
e.g. data transformation for images could be rotation, scaling, varying illumination conditions and so on

Example on overfitting and data augmentation:

this neural network will shape exactly this dataset
error (training) = 0 but maybe it is not good in generalizing



dataset with augmentation; in this case it is more difficult for the network to overfit

The variations should be small.

③ Early stopping

Keep a validation set and monitor the loss. The loss on training set will always decrease but on the validation set it will decrease up to a certain point and then increase.
You can stop iterations early to avoid overfitting when the loss on the validation starts increasing.
When to stop? Use cross-validation to determine best values.

④ Parameter sharing

Overfitting is the ability of the model to perfectly shape the dataset; main causes of this: too many parameters or too many iterations during training phase.

When we have a complex problem it is important to have many layers and to have many parameters in these layers; it is often impossible to model a real problem with too few parameters.

A big network is thus required to model the problem but we may end up with overfitting.

The most convenient solution is not to reduce the parameters (otherwise the modeling power of the network will no more be sufficient), but instead to constrain some of them to be the same. The number of units remains the same but the number of trainable parameters is smaller. The layout of the network stays the same.

This also brings advantages in memory storage.

⑤ Dropout

The problem is the same as before. With dropout we keep the structure but at each step we randomly drop some connections (do not take into account for them during weight update). The connection is not physically removed, it could be considered in the next iterations. At each step, the dropout operator will ignore some of the connections.

Dropout ≠ pruning: in pruning the connections are physically removed

Pruning is the practice of removing parameters (individual parameters or groups of them) from a network. The goal is to maintain the accuracy of the network while increasing its efficiency